REMARKS/ARGUMENTS

Claims 1-19 are pending in the application. Claim 1 is rejected under 35 U.S.C. §101 as

being directed to non-statutory subject matter. Claims 1, 3-8, 12-15 and 17-19 are rejected under

35 U.S.C. §103(a) as being unpatentable over Patel et al., *Improving Trace Cache Effectiveness*

with Branch Promotion and Trace Packing, in further view of Johnson, U.S. Patent No.

5,924,092. Claims 2, 9-11 and 16 are rejected under 35 U.S.C. §103(a) as being unpatentable

over Patel et al., *Improving Trace Cache Effectiveness with Branch Promotion and Trace*

Packing, in further view of Johnson, U.S. Patent No. 5,924,092, in further view of Peled et al.,

U.S. Patent No. 6,076,144. Claim 8 is amended to bring it into bring it into better form.

With regard to the §101 rejection, claims 1-4 are amended. Applicants submit claims 1-4

in their present form overcome the §101 rejection, and the rejection should be withdrawn.

Applicants submit the cited references do not teach, suggest or disclose at least "[a] cache

comprising: a cache line to store an instruction segment further comprising a plurality of

instructions stored in sequential positions of the cache line in reverse program order" (e.g., as

described in claim 1)

The Office Action asserts that it would be obvious to modify the instruction segment of

Patel with the teaching of Johnson in order to store instructions of an instruction trace in reverse

order "so that the frequently accessed and modified head of the trace will be moved and modified

fewer times so that performance is improved." Applicants respectfully disagree. Applicants

submit in order to establish *prima facie* obviousness, there must be some suggestion or

motivation to modify the reference or combine the reference teachings. For at least the

following reasons, there is no such suggestion or motivation here.

Patel discloses the improvement of fetch rates in trace caches by employing branch

promotion and trace packing. Branch promotion removes the overhead resulting from dynamic

branch prediction by applying static branch prediction to strongly biased branches. Trace

packing packs as many instructions as possible into a pending trace so that more instructions

segments may be fetched during a single fetch cycle. However, Patel neither teaches nor

suggests that the fetch rates may be improved by reversing the order of the instructions in the

traces. Applicants submit there would be no motivation to do so, since reversing instruction

order would not appear to improve fetch rates given the teaching of Patel.

By definition, a trace is a sequence of dynamically executed instructions, which may

originally reside in non-continuous portions of the program memory, starting with a single entry

instruction and ending with multiple exit instructions. For a typical trace, the head of the trace,

i.e., the first instruction in a sequence, is followed by the next executable instruction in the

sequence, then the next, and so on. If the Office Action's assertions (discussed above) are to be

believed, the first instruction is accessed and modified more than the second, third, etc.,

instructions. This is contrary to the known operation of the typical trace.

Applicants submit it is unclear how accessing the first instruction in a trace more

frequently than the second instruction, and accessing the second instruction more frequently than

the third, and so on, would improve the performance of a trace (thereby eliminating any

motivation to do so). Since the trace defines sequential instructions, which perform a particular

operation, accessing the instructions in decreasing frequency would in no way advance the

completion of the particular operation. Indeed, such access of the trace would hinder the

completion, thereby defeating the purpose of the trace.

84331.1   -6-

Moreover, it is unclear how modifying the first instruction in a trace more frequently than the second instruction, and modifying the second instruction more frequently than the third, and so on, would improve the performance of a trace. Again, since the trace defines sequential instructions, which perform a particular operation, modifying the instructions in decreasing frequency would in no way advance the completion of the particular operation. Indeed, such modification would result in a different operation, thereby, defeating the purpose of the trace.

Therefore, the Office Action's asserted motivation for modifying Patel with Johnson does not apply.

Furthermore, even if the head of the trace could be more frequently accessed and modified, the Office Action has provided no explanation of how such would improve the fetch rates of the trace cache of Patel.

As stated previously, there is no motivation to reverse the instructions in a Patel trace. Patel discloses using branch promotion to improve cache fetch rates. The purpose of branch promotion is to reduce the dynamic branching of strongly biased traces by applying static branches (or predictions). Applicants submit reversing the instructions so that the first instruction is listed last in the trace does not improve the branch promotion technique. Reversing the instructions does not reduce the dynamic branching of strongly biased traces. Moreover, it does not improve the fetch rates. As such, there is no reason a person of ordinary skill in the art would be motivated to reverse the instructions in a trace, while using branch promotion, to improve fetch rates.

Patel also discloses using trace packing to improve cache fetch rates. The purpose of trace packing is to increase the number of instructions fetched per fetch cycle. However, Applicants submit reversing the instructions so that the first instruction is listed last in the trace

84331.1   -7-

does not improve the trace packing technique. Moreover, such does not appear to improve the fetch rates. As such, a person of ordinary skill in the art would not be motivated to reverse the instructions in a trace, while using trace packing, to improve fetch rates.

Applicants further respectfully disagree with the Office Action's assertion that Johnson has taught that the static sorting algorithm stores elements in reverse order, since it stands to reason that the elements first added to the array would be accessed and used before the elements most recently added to the array. *See* Office Action dated 12/2/2005, page 11, paragraph 43, lines 1-3. Applicants disagree with the assertion and the associated rationales found in paragraph 43. Applicants note the current Office Action does not cite to any specific section of Johnson to support its assertion.

However, previously in discussing Johnson, the Office Action cited column 4, lines 13-24. Column 4, lines 13-24 of Johnson state:

> For the illustrated embodiment discussed below, a static sorting algorithm is used which arranges the logical blocks of data in a logical page in reverse order, thereby placing the last logical block at the beginning of the array, and the first logical block at the end. Consequently, *the more frequent modifications* to the data in the first logical block require recompression and/or moving of fewer, or no other, subsequent frames than the *less frequent modifications* to the data in the last logical block of a page. In general, this results in a lower average number of *updated frames per data modification,* and thus improved overall performance. (*emphasis added*)

The cited section discusses improving overall performance based upon lowering the average number of update frames *per modification* in a data array. The cited section does not, however, discuss improving overall performance during *accessing* or *using* data in a data array. *Accessing* data or *using* data is not the same as *modifying* data, and the cited section of Johnson does not discuss the benefits of its sorting algorithm during data access or data "use" at all.

The Johnson reference is not directed toward improving performance during data access

or use at all, but rather limited to discussing improving performance during data modification.

Nevertheless, the Office Action asserts:

> Johnson has taught that the static sorting algorithm stores elements in reverse order, since it stands to reason that the elements first added *to the array would be accessed and used* before the elements most recently added to the array. (*emphasis added*) *See* Office Action, paragraph 43, lines 1-3.

and

> Johnson has taught that entries that were placed at the beginning of an array, e.g. placed first within the array, *are more likely to be accessed first*, so, by placing these elements at the end of an array where there are less elements dependent on that one particular element to be made when the elements are modified, such as when Patel's instruction segments are fetched in split groups, less time is required to update the element. (*emphasis added*) *See* Office Action, paragraph 43, lines 14-18.

Applicants submit that these overbroad assertions are erroneous and unsupported by the Johnson

reference. Applicants submit the related assertions based upon these interpretations of Johnson

found in paragraph 43 of the recent Office Action are erroneous as well, and insufficient to

support a proper teaching, suggestion or motivation to combine the teachings of Patel and

Johnson. Applicants maintain that a person of ordinary skill in the art would not be motivated to

reverse the instructions in a trace, while using trace packing, to improve fetch rates and that as

such, claim 1 is allowable in its present form. Independent claims 5, 6, 8 and 14 contain similar

allowable limitations, and are therefore allowable for similar reasons. Claims 2-4, 7, 9-13 and

15-19 are allowable for depending from allowable base claims.

B.   Claims 2, 9-11 and 16

The deficiencies are not corrected by Peled. Peled discloses a cache organized around

trace segments of the running programs rather than an organization based on memory addresses.

84331.1   -9-

Applicatio No. 09/708,722   Attorney Docket No. 2207/9800
Amendment dated April 3, 2006
Office Action dated December 2, 2005

However, Peled fails to provide any motivation for modifying Patel with Johnson. Moreover,

there is no motivation disclosed to modify Patel with Johnson and Peled to arrive at the claimed

invention. Accordingly, the Office Action has failed to establish a *prima facie* case of

obviousness over Patel in view of Johnson in further view of Peled.

For at least these reasons, the Claims 1-19 are believed to be patentable over the cited

references, individually and in combination. Withdrawal of the rejections is, therefore,

respectfully requested.

For at least all the above reasons, the Applicant respectfully submits that this application

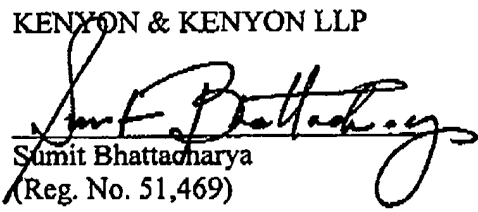is in condition for allowance. A Notice of Allowance is earnestly solicited.

The Examiner is invited to contact the undersigned at (408) 975-7500 to discuss any

matter concerning this application. The Office is hereby authorized to charge any additional fees

or credit any overpayments under 37 C.F.R. § 1.16 or § 1.17 to Deposit Account No. 11-0600.

                                        Respectfully submitted,

                                        KENYON & KENYON LLP

Date: April 3, 2006   By: _____
                                        Sumit Bhattacharya
                                        (Reg. No. 51,469)

KENYON & KENYON LLP
333 West San Carlos St., Suite 600
San Jose, CA 95110
Telephone: (408) 975-7500
Facsimile: (408) 975-7501

84331.1   -10-